BIOSTATS 640 – Intermediate Biostatistics
Spring 2022

Introduction to R
05 – How to Create Variables and Illustration of Multiple Linear Regression

Introduction.

As we are immersed in normal theory regression, this R lesson is about that. In this lesson you will gain practice in some of the steps involved in performing a multiple predictor linear regression. The steps that are not included are about assessing the model assumptions and reporting regression results. We'll tackle these in a subsequent lesson.

This lesson also introduces some basics in how to create variables. One of the important "take aways" is that what we think of as "categorical variables" has a different name in R: factors. A related "take away" that it is important to know how R stores factor variables because their storage has an impact on the analyses we do. Stay tuned for that.

The dataset we will use is a subset of n=1000 observations and 9 variables from the Framingham Heart Study Data. Download the following excel dataset from the course website: **`framingham_1000.xlsx`**.

As needed, see again R lesssons 01 - 04:
01 – R Essentials (R Studio interface, the console as a giant calculator, a few basic commands)
02 – Introduction to Packages and Simple Data Description
03 - Working Directory, R Markdown and Data Inspection
04 - Introduction to **`tidyverse`** and **`dplyr`** and illustration of simple linear regression

Before you begin

__1. Packages to download: **`GGally`**

__2. Download from the course website the dataset framingham_1000.xlsx and place it in your working directory.

1. Introduction to the Framingham Heart Study

framingham_1000.xlsx

Source:
Levy (1999) National Heart Lung and Blood Institute. Center for Bio-Medical Communication.
Framingham Heart Study

Description:

Cardiovascular disease (CVD) is the leading cause of death and serious illness in the United States. In 1948, the Framingham Heart Study - under the direction of the National Heart Institute (now known as the National Heart, Lung, and Blood Institute or NHLBI) was initiated.

The objective of the Framingham Heart Study was to identify the common factors or characteristics that contribute to CVD by following its development over a long period of time in a large group of participants who had not yet developed overt symptoms of CVD or suffered a heart attack or stroke.

In this illustration, we will explore the fitting of a multiple predictor model to systolic blood pressure as the dependent variable (Y=ln_sbp). The predictors of interest will be body mass index and serum cholesterol (ln_bmi, ln_scl). An additional covariate of interest will be sex at birth (sex).

Data dictionary/Codebook

Position	Variable	Label	Type	Codings
1	sex	Sex at birth	numeric	1=male 2=female
2	sbp	Systolic Blood Pressure (mm Hg)	numeric	
3	scl	Serum cholesterol (mg/100 ml)	numeric	
4	age	Age, years	numeric	
5	bmi	Body Mass index (kg/m^2)	numeric	
6	id	Study ID	numeric	
7	ln_bmi	Natural logarithm of bmi	numeric	ln_bmi=ln(bmi)
8	ln_sbp	Natural logarithm of sbp	Numeric	ln_sbp=ln(sbp)
9	ln_scl	Natural logarithm of scl	Numeric	ln_scl=ln(scl)

2. Highlights from Lesson 04
Introduction to `tidyverse`, `dplyr` and illustration of simple linear regression

1. **`tidyverse`** is actually a bundle of packages.
    a. It includes 8 core packages. If you have already issued **`library(tidyverse)`** command, you do not need to issue another **`library( )`** command to access these packages.
    b. It also includes 11 additional packages. To access any of the additional packages, you DO NEED to issue another **`library( )`** command.

2. The pipe operator **`%>%`** is a terrific programming utility.
    a. Translation "AND THEN", the pipe operator allows you to chain together your commands.
    b. For ease of readability, consider putting each chained command on its own line (and comment it!).

3. The **`tidyverse`** ipackage includes among its core packages a package called **`dplyr`**. **`dplyr`** has lots of functions that are wonderful for all sorts of basic data management, including:
    a. **`select( )`** to choose **variables**
    b. **`filter( )`** to choose **observations**
    c. **`mutate( )`** to **create new variables**.

4. **Encouraged.** Develop a structured approach to your R Markdown work

5. **Encouraged.** Code your chunks of executable commands such that each "chunk" performs just one task. This will make troubleshooting, editing, and archiving your work much easier. Hooray.

6. **ALWAYS check your dataset structure** (number of observations, variables, storage, etc.)

7. **ALWAYS LOOK AT YOUR DATA** (especially before modeling, obtain summaries and visualizations.

3. How to Create New Variables
The Basics

Before you begin - Video
Consider watching this video.
(Source: Greg Martin, R Programming 101)
Recode data using R Programming: Use the tidyverse and dplyr to create a new variable (video, 7:04)

3.1 Create a New Numeric Variable From a Numeric variable

```
library(tidyverse)

newdataframe <- olddataframe %>%
                mutate(newvariable = definitionhere)
```

```
# EXAMPLE
temp <- framingham %>%
        mutate(age_days = age*365.2,
               new_lnbmi = log(bmi))
```

3.2 Create a 0/1 Indicator

```
Key

library(tidyverse)

newdataframe <- olddataframe %>%
                mutate(newvariable = ifelse(condition_thatmustbetrue, 1, 0)
```

```
# EXAMPLE
temp <- framingham %>%
        mutate(female01 = ifelse(sex==2,1,0))        # don't forget that this must have double ==
```

3.3 Create a factor variable of groups of a numeric variable

```
Numeric → Character → Factor
library(tidyverse)
newdata <- olddata %>%
   mutate(newvariable = case_when(
                          oldvariable conditiontobetrue ~ "assignment",
                          oldvariable conditiontobetrue ~ "assignment")) %>%
  mutate(newvariable = factor(newvariable,
                      levels = c("name","name")))
```

```
temp <- framingham %>%                                    # TIP do not overwrite original var
   mutate(age_group = case_when(                          # step 1: create character var
                          age %in% 30:39 ~ "30-39",
                          age %in% 40:49 ~ "40-49",
                          age %in% 50:59 ~ "50-59",
                          age %in% 60:69 ~ "60-69",
                          age %in% 70:79 ~ "70-79")) %>%
  mutate(age_group = factor(age_group,                    # step 2: convert character to factor
                      levels = c("30-39","40-49", "50-59", "60-69", "70-79")))

  table(temp$age_group)
```

```
Numeric → Numeric → Factor
Key
library(tidyverse)


newdata <- olddata %>%
   mutate(newvariable = case_when(
                          oldvariable conditiontobetrue ~ number,
                          oldvariable conditiontobetrue ~ number) %>%
  mutate(newvariable = factor(newvariable,
                        levels = c(number, number),
                        levels = c("name","name")))
```

```
temp <- framingham %>%                                  # TIP do not overwrite original var
   mutate(age_play = case_when(                         # step 1: create numeric var
                          age %in% 30:39 ~ 3,
                          age %in% 40:49 ~ 4,
                          age %in% 50:59 ~ 5,
                          age %in% 60:69 ~ 6,
                          age %in% 70:79 ~ 7)) %>%

  mutate(age_play = factor(age_play,                     # step 2: convert number to factor
                      levels=c(3,4,5,6,7),
                      labels = c("30-39","40-49", "50-59", "60-69", "70-79")))

table(temp$age_play)
```

Mathematical Functions

Function	Definition	Example
+	Addition	`> 2+2` `[1] 4`
-	Subtraction	`> 5-3` `[1] 2`
*****	Multiplication	`> 5*4` `[1] 20`
/	Division	`> 20/4` `[1] 5`
^	Exponentiation (raising to a power)	`> 6^2` `[1] 36`
%/%	Integer part of division or quotient	`> 48 %/% 5` What is whole number of 48/5? `[1] 9`
%%	Remainder part of division or quotient	`> 48 %% 5` What is the remainder of 48/5? `[1] 3`
log()	logarithm to base e ("natural log") You may know this as ln( )	`> log(34)` `[1] 3.526361` $e^{3.526361} = 34$
log10()	Logarithm to base 10	`> log10(100)` `[1] 2` $10^2 = 100$
exp()	Exponentiation of the constant e Recall: e = 2.718 … (approx.)	`> exp(4)` `[1] 54.59815` $e^4 = 54.59815$
sqrt()	Square root of	`> sqrt(100)` `[1] 10` $\sqrt{100} = 10$
round(x,n)	Round x to the nth digit	

4. Illustration of Multiple Linear Regresion
Model Estimation

import framingham_1000.xlsx but name it framingham

```
library(readxl)
framingham <- read_excel("framingham_1000.xlsx")
str(framingham)
```

```
## tibble [1,000 × 9] (S3: tbl_df/tbl/data.frame)
##  $ sex   : num [1:1000] 1 1 1 1 1 1 1 1 1 1 ...
##  $ sbp   : num [1:1000] 140 118 132 104 114 150 132 155 152 124 ...
##  $ scl   : num [1:1000] 276 196 155 230 188 234 255 215 165 312 ...
##  $ age   : num [1:1000] 44 36 57 33 42 37 47 46 46 39 ...
##  $ bmi   : num [1:1000] 25.3 22.8 24.5 26.6 21.8 ...
##  $ id    : num [1:1000] 2290 1834 2134 569 1340 ...
##  $ ln_bmi: num [1:1000] 3.23 3.13 3.2 3.28 3.08 ...
##  $ ln_sbp: num [1:1000] 4.94 4.77 4.88 4.64 4.74 ...
##  $ ln_scl: num [1:1000] 5.62 5.28 5.04 5.44 5.24 ...
```

Examples of Creating New Variables

```
library(tidyverse)              # one time:  Attach tidyverse

# create some new numeric vars
temp <- framingham %>%
        mutate(age_days = age*365.2,
               new_lnbmi = log(bmi))
str(temp)                                          # quick peek
```

```
## tibble [1,000 × 11] (S3: tbl_df/tbl/data.frame)
##  $ sex      : num [1:1000] 1 1 1 1 1 1 1 1 1 1 ...
##  $ sbp      : num [1:1000] 140 118 132 104 114 150 132 155 152 124 ...
##  $ scl      : num [1:1000] 276 196 155 230 188 234 255 215 165 312 ...
##  $ age      : num [1:1000] 44 36 57 33 42 37 47 46 46 39 ...
##  $ bmi      : num [1:1000] 25.3 22.8 24.5 26.6 21.8 ...
##  $ id       : num [1:1000] 2290 1834 2134 569 1340 ...
##  $ ln_bmi   : num [1:1000] 3.23 3.13 3.2 3.28 3.08 ...
##  $ ln_sbp   : num [1:1000] 4.94 4.77 4.88 4.64 4.74 ...
##  $ ln_scl   : num [1:1000] 5.62 5.28 5.04 5.44 5.24 ...
##  $ age_days : num [1:1000] 16069 13147 20816 12052 15338 ...
##  $ new_lnbmi: num [1:1000] 3.23 3.13 3.2 3.28 3.08 ...
```

```
# create a 0/1 indicator of female sex
# KEY: ifelse(CONDITION,code_if_true, code_if_false)
temp <- framingham %>%
        mutate(female01 = ifelse(sex==2,1,0))
table(temp$sex,temp$female01)                      # check
```

```
##
##       0   1
##   1 443   0
##   2   0 557
```

```
# create grouped levels of age (numeric → character → factor)


temp <- framingham %>%
   mutate(age_group = case_when(
                           age %in% 30:39 ~ "30-39",
                           age %in% 40:49 ~ "40-49",
                           age %in% 50:59 ~ "50-59",
                           age %in% 60:69 ~ "60-69",
                           age %in% 70:79 ~ "70-79")) %>%
  mutate(age_group = factor(age_group,
                       levels = c("30-39","40-49", "50-59", "60-69", "70-79")))

table(temp$age_group)

##
## 30-39 40-49 50-59 60-69 70-79
##   290   346   294    70     0
```

Check Dataset Structure

```
str(framingham)

## tibble [1,000 × 9] (S3: tbl_df/tbl/data.frame)
##  $ sex   : num [1:1000] 1 1 1 1 1 1 1 1 1 ...
##  $ sbp   : num [1:1000] 140 118 132 104 114 150 132 155 152 124 ...
##  $ scl   : num [1:1000] 276 196 155 230 188 234 255 215 165 312 ...
##  $ age   : num [1:1000] 44 36 57 33 42 37 47 46 46 39 ...
##  $ bmi   : num [1:1000] 25.3 22.8 24.5 26.6 21.8 ...
##  $ id    : num [1:1000] 2290 1834 2134 569 1340 ...
##  $ ln_bmi: num [1:1000] 3.23 3.13 3.2 3.28 3.08 ...
##  $ ln_sbp: num [1:1000] 4.94 4.77 4.88 4.64 4.74 ...
##  $ ln_scl: num [1:1000] 5.62 5.28 5.04 5.44 5.24 ...
```

Always look at your data

```
library(stargazer)

summary(framingham)                                  # no frills but complete

##       sex              sbp              scl              age
##  Min.   :1.000   Min.   : 80.0   Min.   :115.0   Min.   :30.00
##  1st Qu.:1.000   1st Qu.:116.0   1st Qu.:197.0   1st Qu.:38.75
##  Median :2.000   Median :128.0   Median :225.0   Median :45.00
##  Mean   :1.557   Mean   :132.3   Mean   :227.8   Mean   :45.92
##  3rd Qu.:2.000   3rd Qu.:144.0   3rd Qu.:255.0   3rd Qu.:53.00
##  Max.   :2.000   Max.   :270.0   Max.   :493.0   Max.   :66.00
##                                  NA's   :4
##       bmi              id           ln_bmi          ln_sbp          ln_scl
##  Min.   :16.40   Min.   :   1   Min.   :2.797   Min.   :4.382   Min.   :4.745
##  1st Qu.:23.00   1st Qu.:1246   1st Qu.:3.135   1st Qu.:4.754   1st Qu.:5.283
##  Median :25.10   Median :2488   Median :3.223   Median :4.852   Median :5.416
##  Mean   :25.57   Mean   :2410   Mean   :3.230   Mean   :4.872   Mean   :5.410
##  3rd Qu.:27.80   3rd Qu.:3605   3rd Qu.:3.325   3rd Qu.:4.970   3rd Qu.:5.541
##  Max.   :43.40   Max.   :4697   Max.   :3.770   Max.   :5.598   Max.   :6.201
##  NA's   :2                      NA's   :2                       NA's   :4
```

```
framingham <- as.data.frame(framingham)              # compact table
stargazer(framingham, type="text", median=TRUE)

##
## ===============================================================================
## Statistic   N     Mean    St. Dev.    Min   Pctl(25) Median  Pctl(75)   Max
## -------------------------------------------------------------------------------
## sex       1,000   1.557     0.497      1       1        2       2        2
## sbp       1,000  132.350   23.043     80      116      128     144      270
## scl        996   227.846   45.087   115.000 197.000  225.000 255.000  493.000
## age       1,000  45.922     8.545     30      38.8     45       53      66
## bmi        998   25.566     3.848   16.400   23.000  25.100   27.800  43.400
## id        1,000 2,410.031 1,363.439    1    1,246.5  2,487.5 3,605.2   4,697
## ln_bmi     998    3.230     0.147    2.797   3.135    3.223   3.325    3.770
## ln_sbp    1,000   4.872     0.163    4.382   4.754    4.852   4.970    5.598
## ln_scl     996    5.410     0.195    4.745   5.283    5.416   5.541    6.201
## -------------------------------------------------------------------------------
```

Create a 0/1 indicator of female sex at birth and keep only the vars of interest and complete observations only

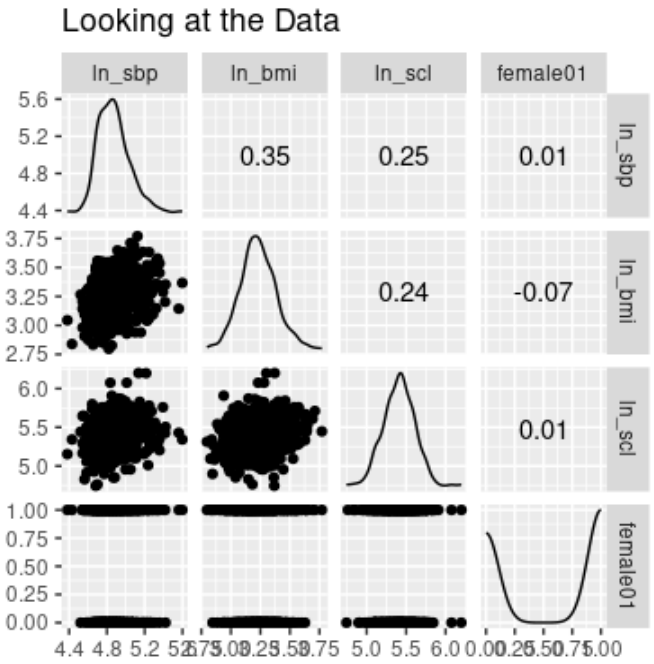
```
library(tidyverse)

mydata <- framingham %>%
        mutate(female01 = ifelse(sex==2,1,0)) %>%
        select(ln_sbp,ln_bmi,ln_scl,female01) %>%
        na.omit()

str(mydata)

## 'data.frame':    994 obs. of  4 variables:
##  $ ln_sbp  : num  4.94 4.77 4.88 4.64 4.74 ...
##  $ ln_bmi  : num  3.23 3.13 3.2 3.28 3.08 ...
##  $ ln_scl  : num  5.62 5.28 5.04 5.44 5.24 ...
##  $ female01: num  0 0 0 0 0 0 0 0 0 0 ...
##  - attr(*, "na.action")= 'omit' Named int [1:6] 279 312 443 491 631 780
##   ..- attr(*, "names")= chr [1:6] "279" "312" "443" "491" ...
```

Pairwise correlations

```
cor(mydata)

##             ln_sbp      ln_bmi      ln_scl     female01
## ln_sbp   1.00000000  0.35083221 0.252434963  0.011905758
## ln_bmi   0.35083221  1.00000000 0.235819016 -0.068910375
## ln_scl   0.25243496  0.23581902 1.000000000  0.009526961
## female01 0.01190576 -0.06891038 0.009526961  1.000000000
```

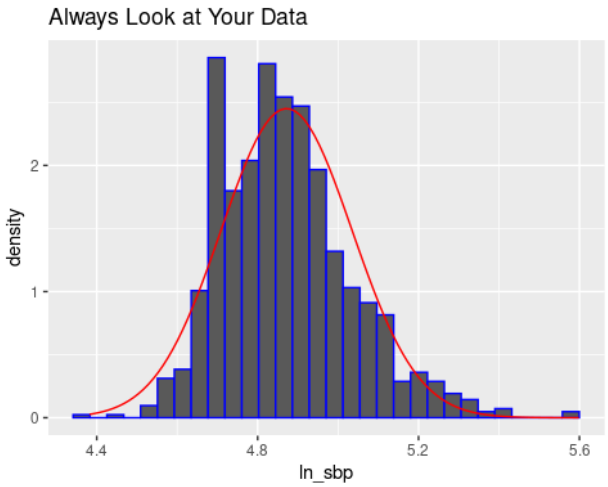
Nicer look: Matrix Plot of Scatter and Correlations using {GGally}

```
library(GGally)

ggscatmat(mydata,columns=c("ln_sbp", "ln_bmi", "ln_scl", "female01")) +
  ggtitle("Looking at the Data")
```



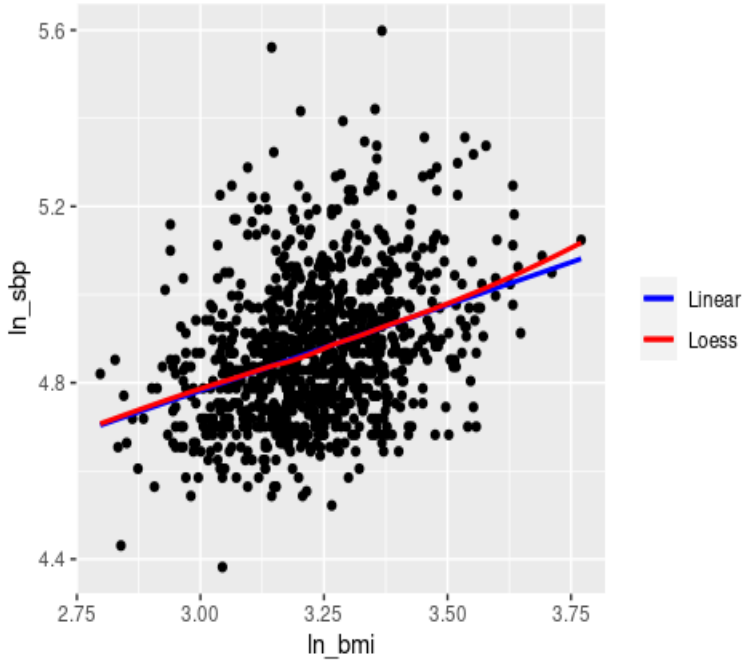
Assessment of Normality of Y=ln_sbp

```
library(ggplot2)

shapiro.test(mydata$ln_sbp)              # Null:  Normality can be assumed

##
##  Shapiro-Wilk normality test
##
## data:  mydata$ln_sbp
## W = 0.9737, p-value = 0.000000000001966

ggplot(data=mydata) +
       aes(x=ln_sbp) +

       geom_histogram(color="blue",
                      aes(y=..density..)) +

       stat_function(fun=dnorm,
                     color="red",
                     args=list(mean=mean(mydata$ln_sbp),
                               sd=sd(mydata$ln_sbp))) +

       ggtitle("Always Look at Your Data")
```



Graphical Assessment of Normality w line and loess

```
library(ggplot2)

ggplot(data=mydata) +
  aes(y=ln_sbp) +
  aes(x=ln_bmi) +

  geom_point( )  +

  geom_smooth(method="lm",
              aes(color="Linear"),
              se=FALSE) +

   geom_smooth(method="loess",
              aes(color="Loess"),
              se=FALSE) +

   scale_colour_manual(name="",values=c("blue", "red"))
```



Fit one predictor models. Summarize

```
library(stargazer)

m1 <- lm(data=mydata, ln_sbp ~ ln_bmi)
m2 <- lm(data=mydata, ln_sbp ~ ln_scl)
m3 <- lm(data=mydata, ln_sbp ~ female01)

stargazer(m1,m2,m3,
          type="text",
          font.size="small",
          alight=TRUE,
          omit.stat=c("f","ser"))

##
## ==========================================
##                   Dependent variable:
##               ---------------------------
##                          ln_sbp
##                 (1)       (2)       (3)
## ------------------------------------------
## ln_bmi       0.388***
##               (0.033)
##
## ln_scl                  0.211***
##                          (0.026)
##
## female01                             0.004
##                                     (0.010)
##
## Constant     3.618***  3.730***  4.870***
##               (0.106)   (0.139)   (0.008)
##
## ------------------------------------------
## Observations    994       994       994
## R2             0.123     0.064    0.0001
## Adjusted R2    0.122     0.063    -0.001
## ==========================================
## Note:          *p<0.1; **p<0.05; ***p<0.01
```

Fit two multiple predictor models and do a partial F test

```
library(stargazer)

mfull <- lm(data=mydata, ln_sbp ~ ln_bmi + ln_scl)
m1_bmi <- lm(data=mydata, ln_sbp ~ ln_bmi)
m1_scl <- lm(data=mydata, ln_sbp ~ ln_scl)

# Partial F:  GIVEN ln_scl is ADDITION of ln_bmi statistically significant?
cat("\nPartial F:  Null: Given ln_scl, ln_bmi is NOT significant\n")

##
## Partial F:  Null: Given ln_scl, ln_bmi is NOT significant
anova(m1_scl,mfull)
## Analysis of Variance Table
##
## Model 1: ln_sbp ~ ln_scl
## Model 2: ln_sbp ~ ln_bmi + ln_scl
##   Res.Df    RSS Df Sum of Sq     F               Pr(>F)
## 1    992 24.641
## 2    991 22.276  1    2.3648 105.2 < 0.00000000000000022 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Partial F:  GIVEN ln_bmi is ADDITION of ln_scl statistically significant?
cat("\nPartial F:  Null: Given ln_bmi, ln_scl is NOT significant\n")
##
## Partial F:  Null: Given ln_bmi, ln_scl is NOT significant

anova(m1_bmi,mfull)

## Analysis of Variance Table
##
## Model 1: ln_sbp ~ ln_bmi
## Model 2: ln_sbp ~ ln_bmi + ln_scl
##   Res.Df    RSS Df Sum of Sq      F         Pr(>F)
## 1    992 23.078
## 2    991 22.276  1   0.80255 35.703 0.000000003201 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Last pretty table

```
stargazer(m1_scl, m1_bmi, mfull,
          type="text",
          font.size="small",
          alight=TRUE,
          omit.stat=c("f","ser"))

##
## ==========================================
##                   Dependent variable:
##               ---------------------------
##                          ln_sbp
##                 (1)       (2)       (3)
## ------------------------------------------
## ln_scl       0.211***            0.150***
##               (0.026)            (0.025)
##
## ln_bmi                  0.388***  0.341***
##                          (0.033)   (0.033)
##
## Constant     3.730***  3.618***  2.956***
##               (0.139)   (0.106)   (0.152)
##
## ------------------------------------------
## Observations    994       994       994
## R2             0.064     0.123     0.154
## Adjusted R2    0.063     0.122     0.152
## ==========================================
## Note:          *p<0.1; **p<0.05; ***p<0.01
```